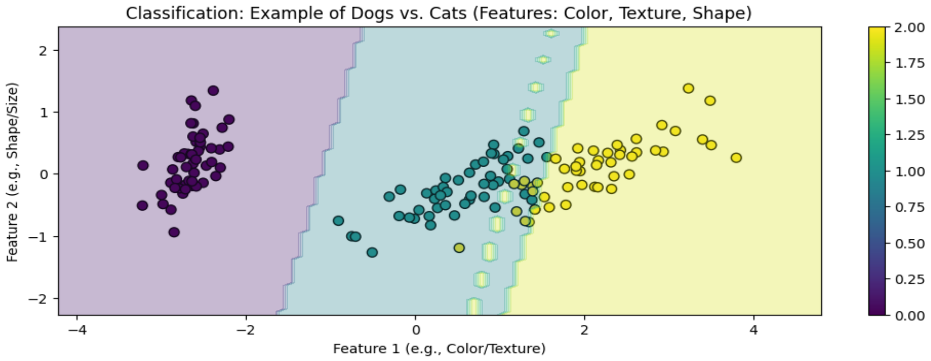
1. CLASSIFICATION :

Classification is one type of Supervised Learning Technique which teaches a machines/model to **sort things into categories** . It learns by looking at examples with labels(like emails are marked 'spam' or 'not spam').

After Learning, it can decide which category new item belongs to, like identifying if a new email is spam or not. For example a classification model might be trained on dataset of images labelled as dogs or cats and it can be used to predict the class of new and unseen images as dogs or cats based on their features as color, texture and shape.



- Each colored dot in the plot represents an individual image,with the color indicating whether the model predicts the image to be dog or cat.

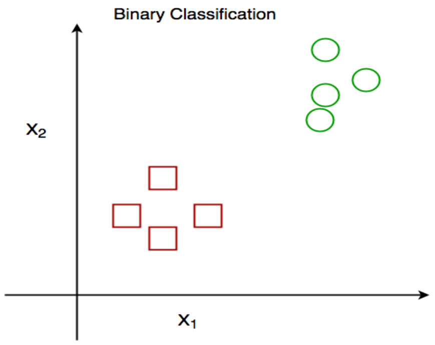
TYPES OF CLASSIFICATION:

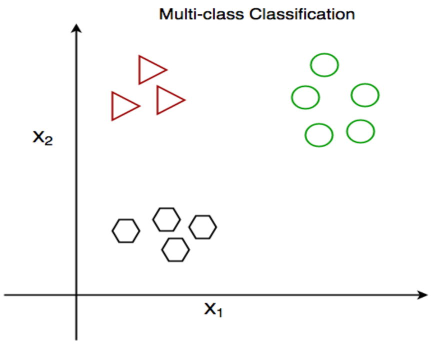
1.BINARY CLASSIFICATION:

This is the simplest type of classification the goal is to sort the data into **two distinct categories**. Think of it like simple choice between two options. Imagine a system that sorts email into spam or not spam. It works by looking at different features of the email like certain keywords or sender details and decides whether it's spam or not spam. It only chooses between two options.

2.**MULTICLASS CLASSIFICATION:**

Instead of two categories, the data needs to be **sorted into more than two categories**. The model picks the one that best matches the input. Think of image recognition system that sorts pictures of animals into categories cat, dog and bird.





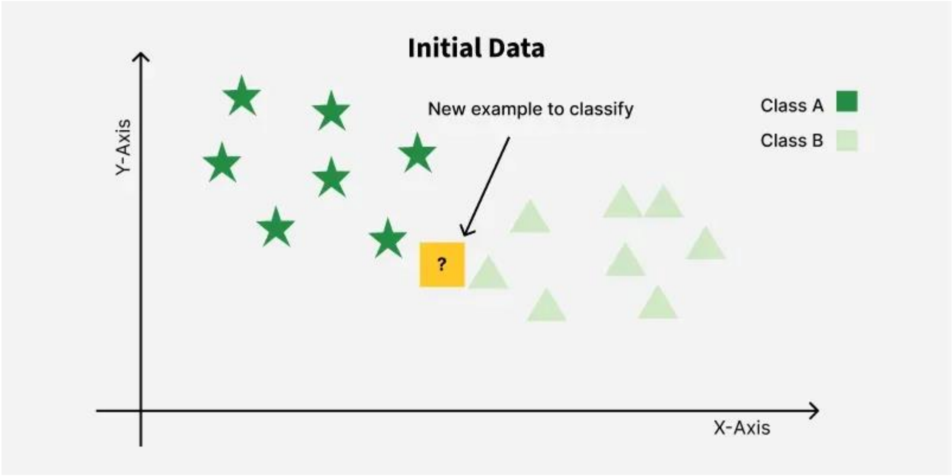
3.Multi-Label classification:

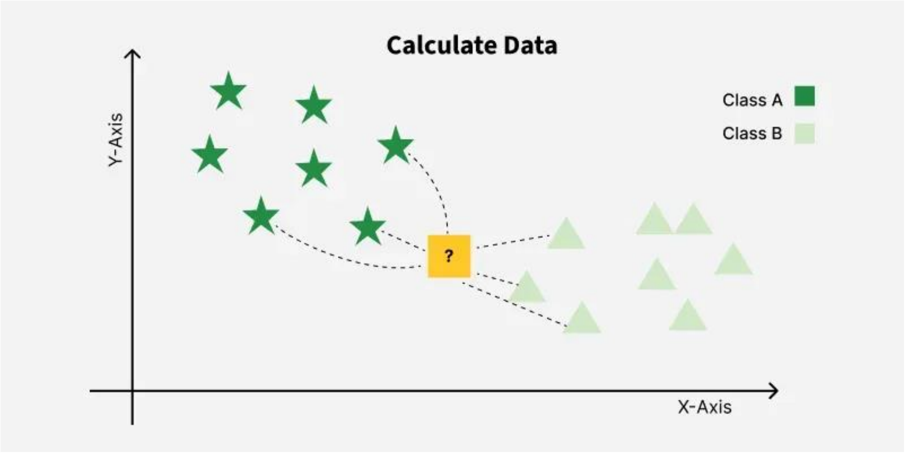
In multi-label classification **single piece of data can belong to multiple categories at once**. Unlike multiclass classification where each data point belongs to only one class, multilabel classification allows datapoints to belong to multiple classes. A movie Recommendation system could tag a movie as both action and comedy. The system checks various features(like movie plot, actors or genre tags) and assign multiple labels to a single piece of data, rather than just one.

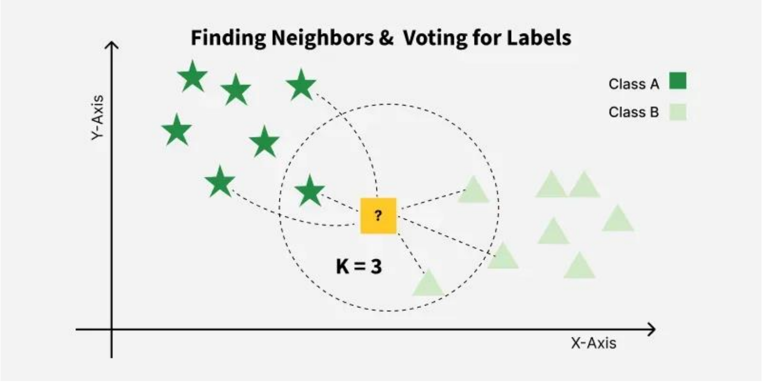
ALOGITHMS IN CLASSIFICATION:

1. K-NEAREST NEIGHBORS(KNN):

K-NEAREST NEIGHBORS(KNN) is a supervised machine learning algorithm generally used for classification but can also used for regression tasks. It works by finding the "K" closest data points(neighbours) to given input and makes a prediction based on majority on the majority class.







KNN is also called a **lazy learner algorithm** because it doesn't learn from the training set immediately **instead it stores the dataset** and at the time of classification it performs an action on the dataset.

What is 'K' in K Nearest Neighbour ?

In the KNN algorithm Kis just a number that tells the algorithm how many nearby points or neighbors to look at when makes a decision.

Example: Imagine you're deciding which fruit it is based on its shape and size. You compare it to fruits you already know.

IF k=3, the algorithm looks at the **3 closest fruits** to the new one.

2 of those 3 fruits are apples and 1 is banana ,the algorithm says the new fruit is an apple because most of its neighbours are apples.

- ## Distance Metrics Used in KNN Algorithm
- ### 1. Euclidean Distance

- Euclidean distance is defined as the straight-line distance between two points in a plane or space. You can think of it like the shortest path you would walk if you were to go directly from one point to another.
  Distance(x,y)= $\sum j=1d(xj-Xij)2$

Step 1: Selecting the optimal value of K

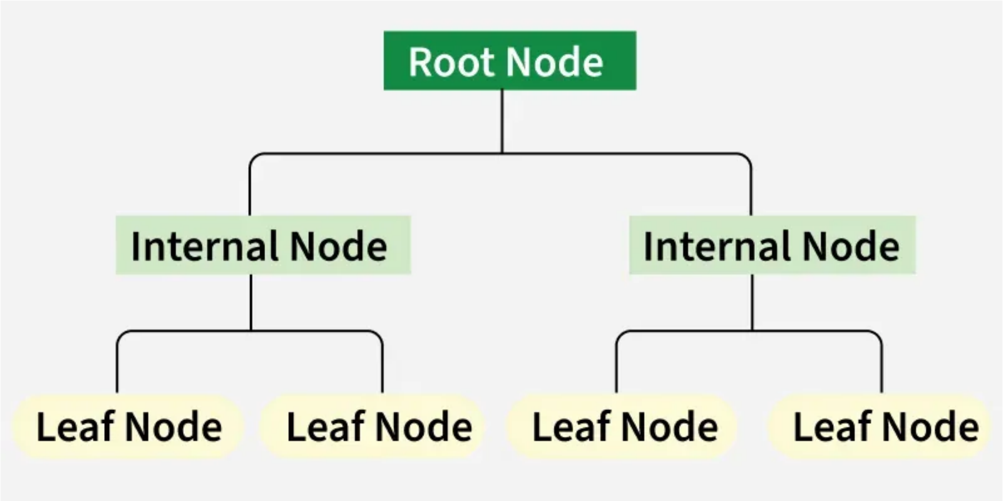
Step 2: Calculating distance

Step 3: Finding Nearest Neighbors

Step 4: Voting for Classification or Taking Average for Regression.

DECISION TREE:

A Decision tree helps us make decision by showing **the different options and how they are related**. It has **a tree-like structure** that starts with one main question called as root node which represents the entire dataset. From there, the tree branches out into different possibilities based on features int the data**.**

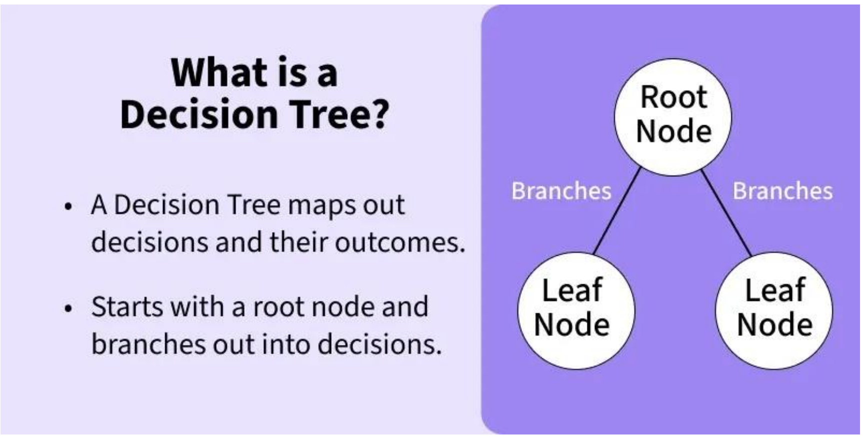


ROOT NODE: starting point representing the whole dataset.

INTERNAL NODES: points where decision are made based on data features.

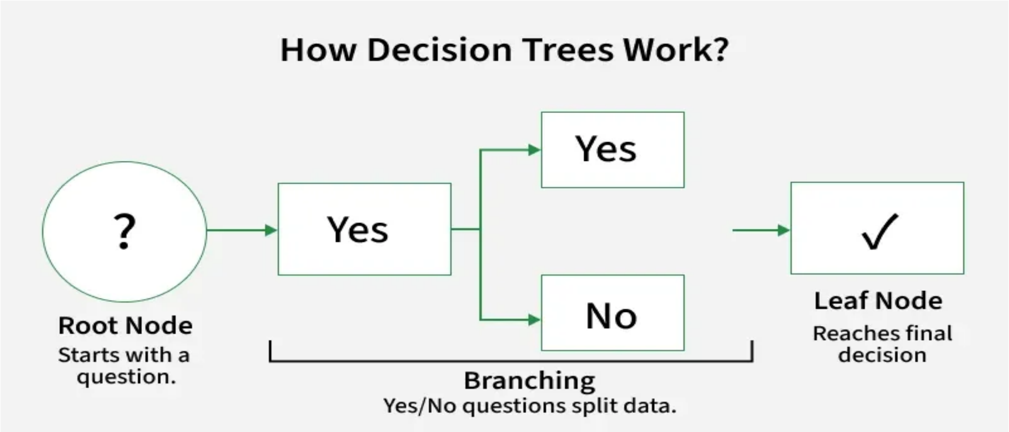
LEAF NODES : End points of the tree where the final decision or prediction is made.

BRANCHES: lines connecting nodes showing the flow from one decision to another.



What is a Decision Tree?

- A Decision Tree maps out decisions and their outcomes.
- Starts with a root node and branches out into decisions.

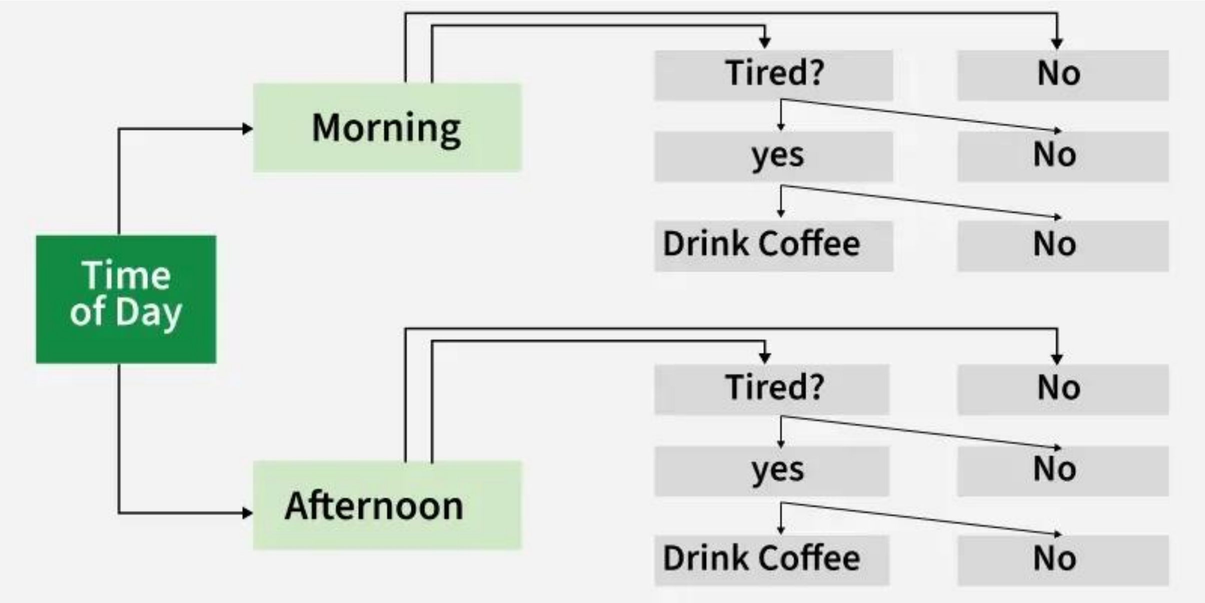


How Decision Trees Work?

Root Node
Starts with a question.

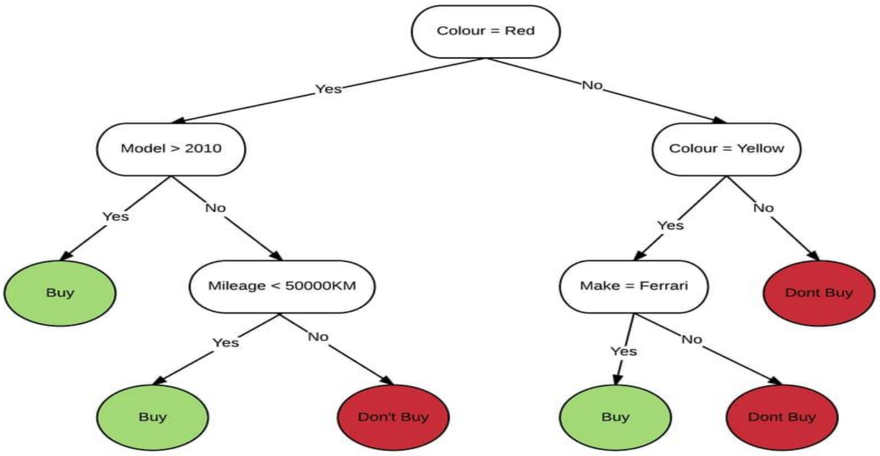
Branching
Yes/No questions split data.

Leaf Node
Reaches final decision



HOW DECISION TREE WORKS:

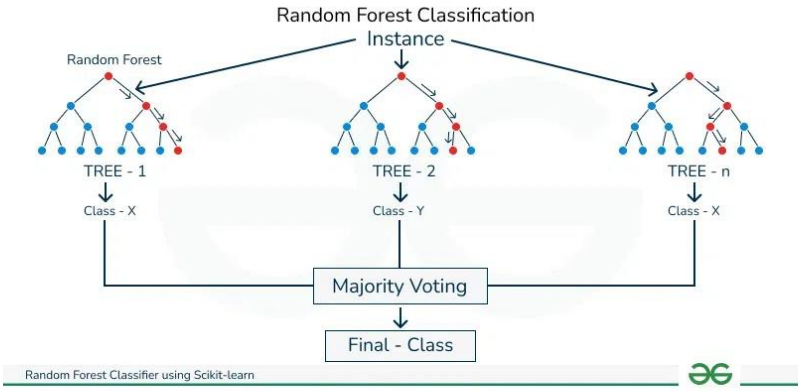
1. **Start with root node**: It begins with a main question at the root node which is derived from the dataset's features.
2. **ASK YES/NO questions**: From the root node the tree asks a series of yes/no questions to split the data into subsets based on specific features.
3. **Branching based on Answers**: Each question leads to different branches:
   If the answer is yes it follows one path
   If the answer is No,it follows another path
4. **Continue Splitting**: This branching continues through further decisions helps in reducing the data down step-by-step
5. **Reach the leaf Node**: The process will end when there are no more question to ask leading to leaf node where the final decision or prediction is made.

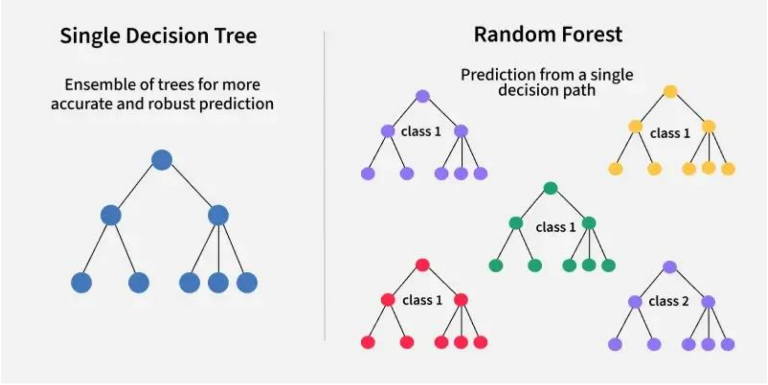


Random Forest:

Is a method that combines the **predictions of multiple decision trees** to produce a more accurate result/output. It can be used for both classification and regression tasks.

In classification tasks, Random forest classification predicts categorical outcomes based on the input data. It uses multiple decisions trees and outputs the labels that **has maximum votes among all individual tree predictions**.




Single Decision Tree
Ensemble of trees for more accurate and robust prediction
Random Forest
Prediction from a single decision path
class 1
class 1
class 1
class 1
class 2

Working of Random Forest Algorithm:

Create many decision tree: The algorithm makes many decision tress each using a random part of the data. So every tree Is bit different.

Pick Random features : When building each tree it **doesn't look at all the features at once**. It picks a few at random to decide how to split the data. This helps the trees stay different from each other.

Each tree make a prediction : Every tree gives its own answer or prediction based on what it learned .